根据：

周转时间=作业完成时间-到达时间

带权周转时间=周转时间/服务时间

平均周转时间=作业周转总时间/作业个数

平均带权周转时间=带权周转时间/作业个数

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 进程名 | A | B | C | D | E | 平均时间 |
|  | 到达时间 | 0 | 1 | 2 | 3 | 4 |  |
|  | 服务时间 | 4 | 3 | 5 | 2 | 4 |  |
| 先来先服务算法（FCFS） | 完成时间 | 4 | 7 | 12 | 14 | 18 |  |
|  | 周转时间 | 4 | 6 | 10 | 11 | 14 | 9 |
|  | 带权周转时间 | 1 | 2 | 2 | 5.5 | 3.5 | 2.8 |
| 短作业优先算法（SJF） | 完成时间 | 4 | 9 | 18 | 6 | 13 |  |
|  | 周转时间 | 4 | 8 | 16 | 3 | 9 | 8 |
|  | 带权周转时间 | 1 | 2.67 | 3.2 | 1.5 | 2.25 | 2.1 |
| 最高响应比优先调度算法（HRRN) | 完成时间 | 4 | 7 | 14 | 9 | 18 |  |
|  | 周转时间 | 4 | 6 | 12 | 6 | 14 | 8.4 |
|  | 带权周转时间 | 1 | 2 | 2.4 | 3 | 3.5 | 2.38 |

# **先来先服务(FCFS)调度算法**是一种最简单的调度算法，该算法既可用于作业调度，也可用于进程调度。当在作业调度中采用该算法时，每次调度都是从后备作业队列中选择一个或多个最先进入该队列的作业，将它们调入内存，为它们分配资源、创建进程，然后放入就绪队列。在进程调度中采用FCFS算法时，则每次调度是从就绪队列中选择一个最先进入该队列的进程，为之分配处理机，使之投入运行。该进程一直运行到完成或发生某事件而阻塞后才放弃处理机。按作业或进程到达的先后顺序进行调度，即每次在后备作业（就绪进程）队列中选择先到达的作业（或进程）投入运行。

CPU繁忙型作业是指该类作业需要大量的CPU时间进行计算，而很少请求I/O。

I/O繁忙型作业是指CPU进行处理时需频繁地请求I/O。

FCFS调度算法有利于CPU繁忙型的作业，而不利于I/O繁忙型的作业(进程)。

**短作业优先调度算法（SJF）**，是指对短作业优先调度的算法。短作业优先(SJF)的调度算法是从后备队列中选择一个或若干个估计运行时间最短的作业，将它们调入内存运行。按作业估计运行时间长短来组织后备作业队列，每次选择运行时间最短的作业投入运行，目的是为了提高系统的吞吐率，减少进程的平均周转时间。后来的作业不抢占正在执行的作业。

采用SJF算法后，不论是平均周转时间还是平均带权周转时间，都有较明显的改善，尤其是对短作业D，其周转时间由原来的11降为3；而平均带权周转时间是从5.5降到1.5。这说明SJF调度算法能有效地降低作业的平均等待时间，提高系统吞吐量。

但是SJF调度算法也有缺点：该算法对长作业不利，如作业C的周转时间由10增至16，其带权周转时间由2增至3.1。更严重的是，如果有一长作业进入系统的后备队列(就绪队列)，由于调度程序总是优先调度那些(即使是后进来的)短作业，将导致长作业长期不被调度；该算法完全未考虑作业的紧迫程度，因而不能保证紧迫性作业会被及时处理；由于作业的长短只是根据用户所提供的估计执行时间而定的，而用户又可能会有意或无意地缩短其作业的估计运行时间，致使该算法不一定能真正做到短作业优先调度。

**HRRN算法**同时兼顾每个作业的等待时间和运行时间两个方面的因素，每次在作业后备队列中挑选响应比最高的作业投入运行。其中响应比计算公式如下：

响应比=（等待时间+要求服务时间）/要求服务时间 = 响应时间/要求服务时间。

如果作业的等待时间相同，则要求服务的时间越短，其优先权越高，有利于短作业。当要求服务的时间相同时，作业的优先权决定于其等待时间，等待时间越长的进程，其优先权越高，因而它实现了先来先服务；对于长作业，作业的优先级可以随等待时间的增加而提高，当其等待时间足够长时，进程的优先级便可升到很高，从而也可获得处理机。